Security Classification: Unclassified

Final Report

Submission Date: October 28, 2016

NeuroCognitive Patterns Phase II Option II

(Aptima Job #1943)

Period of Performance: September 30, 2015 – October 29, 2016

Prepared for:
Dr. Peter Squire, Code 30
875 N. Randolph Street
Arlington, VA 22203-1995
Peter.squire@navy.mil

Prepared by:
Dr. Webb Stacy
Dr. Danielle Ward
Aptima, Inc
12 Gil Street #1400
Woburn, MA 01801
wstacy@aptima.com
dward@aptima.com

Distribution Statement A: Approved for public release: distribution unlimited.

Contract No.: N00014-12-G-0546 0013
Contractor Name: Aptima, Inc.
Contractor Address: 12 Gil Street, Suite 1400, Woburn, MA 01801

In accordance with DFARS 252.227-7013, Aptima, Inc. provides unlimited data rights to "Neurocognitive Patterns II Option II" funded under Contract N00014-12-G-0546 0013, topic ONR BAA15-001.

Prepared and distributed by Terry Green, Contracts Administrator.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Service, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
28-10-2016	Final	30-09-2015-29-10-2016

4. TITLE AND SUBTITLE
NeuroCognitive Patterns

5a. CONTRACT NUMBER
N00014-12-G-0546 0013

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Ward, Danielle, PhD
Stacy, Webb, PhD
Lucia, Lisa, PhD
Katyal, Kapil, PhD
Bennett, Joseph

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Aptima, Inc.
12 Gill Street Suite 1400
Woburn, MA
01801

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research

10. SPONSOR/MONITOR'S ACRONYM(S)
ONR

11. SPONSORING/MONITORING AGENCY REPORT NUMBER

12. DISTRIBUTION AVAILABILITY STATEMENT
Distribution A. Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES **In accordance with DFARS 252.227-7013, Aptima, Inc. provides unlimited data rights to "Neurocognitive Patterns II Option II" funded under Contract N00014-12-G-0546 0013, topic ONR BAA15-001 .**

14. ABSTRACT
Neurocognitive Patterns (NCP) is a new type of neurocognitive architecture, based on an interaction paradigm that mirrors how humans naturally interact with each other and their environment. NCP will provide an innovative brain-computer interface (BCI) for a prosthetic arm that combines recent advances in identifying the neural signals of motor intent with technology that exploits environmental and contextual information to provide likely interpretations for those neural signals. Innovative models for event understanding, object detection, and classification were needed in order to help define the environment. Essentially, the interface will detect and act on user intent. This will result in a new generation of prosthetics that will require dramatically less attention from the operator, freeing the operator to attend to other mission-important tasks.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Danielle Ward
Unclass	Unclass	Unclass	U	25	19b. TELEPONE NUMBER (*Include area code*) 781-496-2479

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI-Std Z39-18

Table of Contents

Executive Summary

Neurocognitive Patterns (NCP) is a new type of neurocognitive architecture, based on an interaction paradigm that mirrors how humans naturally interact with each other and their environment. NCP will provide an innovative brain-computer interface (BCI) for a prosthetic arm that combines recent advances in identifying the neural signals of motor intent with technology that exploits environmental and contextual information to provide likely interpretations for those neural signals. Innovative models for event understanding, object detection, and classification were needed in order to help define the environment. Essentially, the interface will detect and act on user intent. This will result in a new generation of prosthetics that will require dramatically less attention from the operator, freeing the operator to attend to other mission-important tasks.

NeuroCognitive Patterns aims to use neural signals to help to infer motion intent of prosthetic users. While the neural signals are vital to this architecture, contextual and environmental information is also needed in order to best anticipate what action is intended. In order to obtain that contextual and environmental information, a commercially available Kinect Sensor is used to capture video and depth information about an environment. This information is registered into real-space using the Freenect Robot Operating System (ROS) package. This package creates a point cloud from the depth information from the Kinect's video camera, which can then be used to identify and track objects that are in the video. Once an object's location has been identified in the scene and the object has been classified, information about the types of actions that a user may want to make can be inferred using our state machine event understanding module. The output of the state machine is the most likely next action. The NCP system next looks for an intent signal in our EEG processed data. If an intent is present, the most likely next action is passed along to our robotic arm which performs that action.

Project Objective

Brain-Computer interfaces (BCIs) have progressed significantly over the last several years, but two critical challenges still remain. One critical challenge is to develop BCIs that will allow for direct interpretation of users' intent from neural data gathered through non-invasive means. A second critical challenge is that current BCIs often require users to focus all their attention on the system in order to respond to system – generated cues which then result in generation of neural signals that can be time stamped and utilized to infer users' intent.

Neurocognitive Patterns is the first step in providing an innovative brain-computer interface (BCI) for a prosthetic arm that combines recent advances in identifying the neural signals of motor intent with technology that exploits environmental and contextual information to provide likely interpretations for those neural signals. Essentially, the interface detects and act on user intent. This will result in a new generation of prosthetics that will require dramatically less attention and specialized effort from the operator, freeing the operator to attend to other mission-important tasks.

Project Approach

NeuroCognitive Patterns aims to use neural signals to help to infer motion intent of prosthetic users. While the neural signals are vital to this architecture, contextual and environmental information is also needed in order to best anticipate what action is intended. In order to obtain that contextual and environmental information, a commercially available Kinect Sensor is used to capture video and depth information about an environment. This creates a point cloud from the depth information from the Kinect's video camera, which can then be used to identify and track objects in the video. Object motions are candidate stimuli for an event understanding system. As the stimuli arrive, the event understanding system tracks progress against the current event and begins a new event when necessary; as a result, NCP always can report the current status of an event and can predict what stimuli and actions will come next, and this in turn helps interpret the intent signals from the EEG processing module.

In order to demonstrate the utility of NeuroCognitive Patterns (NCP), we needed to collect data and build a system that would help to showcase the different components of our solution. The data required for our solution included EEG data collected from participants performing simple everyday actions. We also required video and location data of the objects in the environment that our participants were interacting with. Finally, we needed to provide the system with some 'knowledge' about the objects that it was to encounter and what actions these objects would likely be involved in.

Architecture

The NeuroCognitive Patterns system was set up such that there are distinct modules that are responsible for different functions of the overall system. These modules include: object

recognition and tracking, event understanding, EEG Analysis, and action. Each of these modules was constructed using the Robot Operating System (ROS) architecture, allowing information to pass between the modules by use of messages. ROS and each of the modules are explained in more detail below in Figure 1.

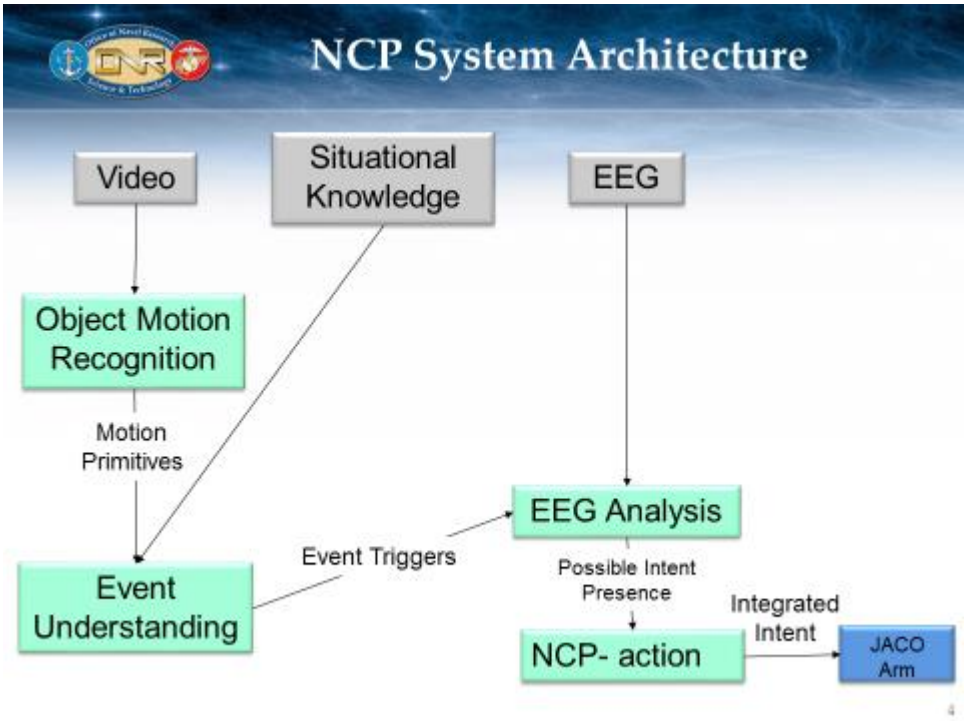

Figure 1: The NCP System architecture diagram with the data coming from the gray boxes, analysis in the green boxes, and action in the blue box.

ROS

The Robot Operating System (ROS) is a flexible framework for writing robot software. It's probably the most common OS used by robotics researchers--it started life in the Stanford AI lab, is overseen by Willow Garage [1] and has been strongly backed by lots of organizations including Google and DARPA. It describes itself as "a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms." Aptima utilized many of the object recognition tools already available in ROS to easily build our object recognition capability into NeuroCognitive Patterns.

Data Collection

Volunteers were seated in a comfortable chair in a quiet room positioned in front of a table with props for completing the every-day tasks of making a cup of coffee and toasting a piece of bread (e.g., coffee maker, toaster, k-cup, mug, bread slice, plate). More specifically, making a cup of coffee involved the following actions: pick up coffee cup, put coffee cup under Keurig machine, pick up K-cup, put K-cup into machine, close pod door, push Keurig machine start button;

toasting a piece of bread involved these actions: pick up piece of bread, turn bread to orient in direction of toaster, place bread into toaster, press toaster button, take toast out of toaster, put toast on plate. Individuals were asked to perform these two tasks five times each as directed by project team members: (a) naturally and without specific direction (i.e., complete the two tasks implementing actions in any order and at their own natural pace), (b) without specific direction but with required pauses in between actions (i.e., implement actions in any order but pausing, resting their hand on the table for an independently determined time, in between actions), (c) naturally but with specific direction (i.e., complete tasks at own pace but following a specifically detailed set of actions), (d) with specific direction and with required pauses (i.e., complete tasks by following a specifically detailed set of actions and pausing in between each action). A Kinect camera was positioned behind volunteers to record actions during these two specific tasks. A 64-channel EEG cap was applied to each person's head to record electrical brain activity during task enactment as well.

EEG system

Brain Vision's ActiCHamp EEG system was used for recording electrical brain activity throughout the session. The cap included 64 Ag/AgCl electrode channels arranged in the standard 10-20 position across the scalp (see Figure 2 below). During recording, an electrode positioned on the left mastoid (behind the left ear) was used as the reference site for scalp electrodes, and a ground electrode was positioned toward the front of the head. Two sets of bipolar electrodes were positioned above and below the right eye and on the left and right temples to record both horizontal and vertical electrooculography (EOG); these were grounded separately with electrodes on both the left and right collarbones. EEG and EOG was recorded at a 500 Hz sampling rate (every 2 milliseconds).

After the recording session, each dataset was filtered offline with a high-pass filter set at 0.1 Hz through the use of BrainVision Analyzer 2.1 software. Events such as action starts and stops were tagged in the EEG data according to an intense analysis of Kinect recorded video files. Lastly, per run, an event marker indicating non-action was tagged as occurring midway between the longest gap between actions.

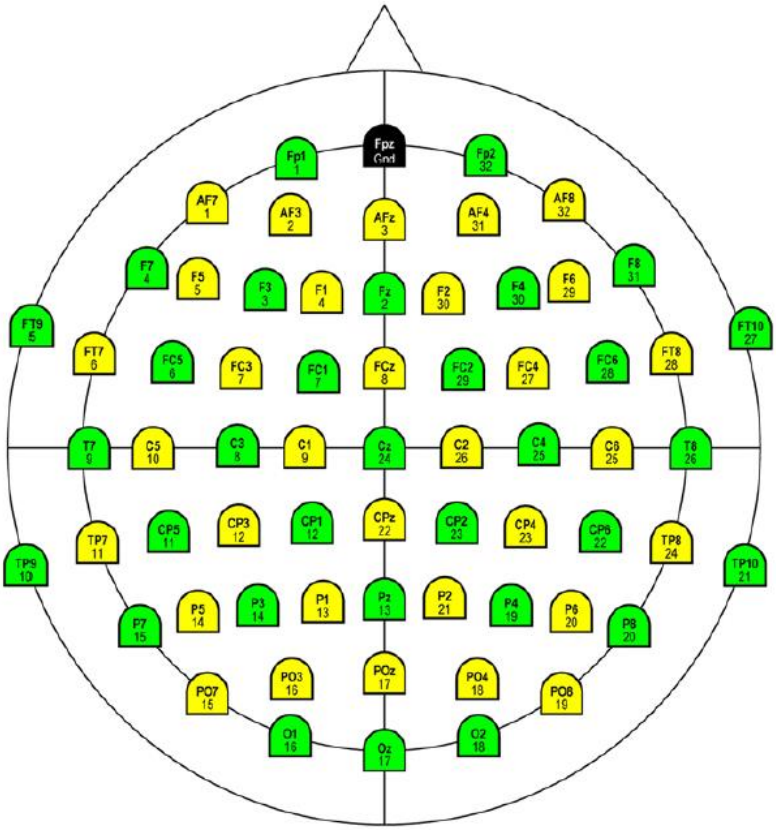

Figure 2: Placement of the 64 electrodes on our EEG cap.

Object Recognition

The Kinect is an RGB-D device that allows you to model the field of view of the Kinect in 3D space and when connected to ROS, a cloud of (red, green, blue, depth) points can be captured for each video frame and recorded in a ROS bag file. This point cloud data serves as an input to the Object Recognition Module in which objects can be identified and tracked.

Object Identification

For our scenario, the objects that need to be identified are:

- the desk
- the plate
- the toast
- the toaster
- the coffee cup
- the k-cup
- the Keurig machine
- the participant's hand

The first step in identifying objects is to locate the plane of space that runs through the surface of the desk in which all the objects of interest are resting. Once this plane is known, points that are near it can be taken out and the remaining sub-clouds (objects of interest) can be determined via clustering.

Desk Location

To discover the desk plane, the RANSAC algorithm is used. RANSAC recursively generates model parameters by testing randomly sampled points from the space to see if they fit the proposed model, and if a sufficient number of points do not fit, the model is refined by repeating the process. In our case, the model is a plane and the sampled points come from the point cloud produced by the Kinect. In order to assist the planar discovery, a rotation matrix is applied to the scene to get the view of the desk in the camera frame to be parallel to the floor, and a thinly sliced z-plane of depth points were given to the RANSAC algorithm. Once a suitable plane is found, each point is scrutinized and rejected if its distance to the plane falls below a threshold.

Object Clustering

Once the plane of the desk has been identified and removed, the remaining points above the desk are put into a Nearest Neighbor algorithm [2] that groups similar points together into object clusters. The clustering algorithm begins by randomly selecting a point and finding its k-nearest neighbors. For each of those k-nearest neighbors, their k-nearest neighbors are found. This process continues until no new nearest neighbors can be added to the object cluster. The end result is a set of sub clouds each of which represents a specific object. After the objects have been clustered, there is a manual step where each object has to be associated with its identity e.g. cluster 1 is a coffee cup, cluster 2 is a piece of bread etc. (Figure 3) Some future work would be aimed at automating this process.

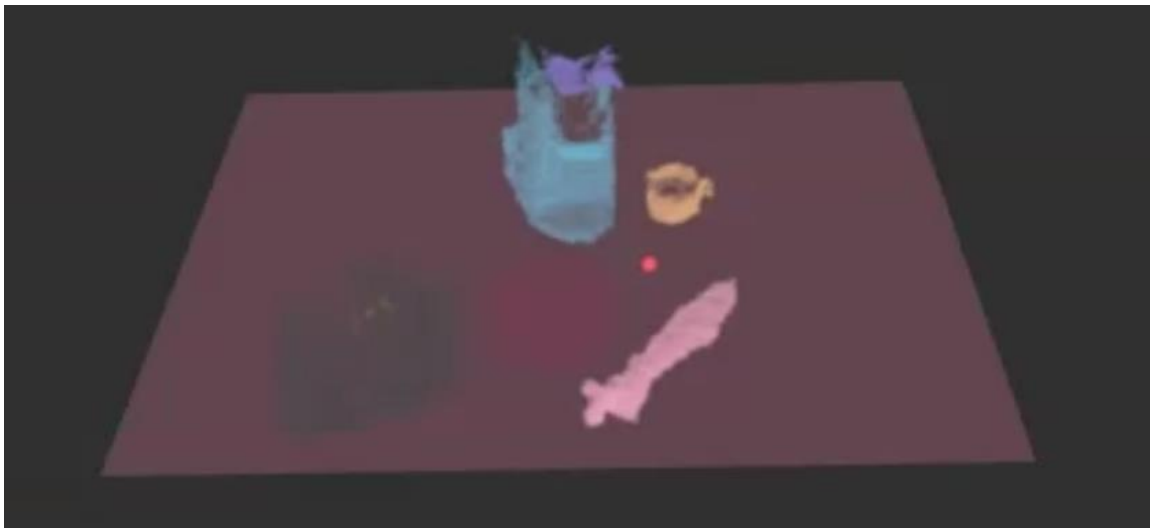

Figure 3: Output of Object Clustering where the different pixels are clustered together into distinct objects and then colored on the screen. In this image, we can see the toaster as a dark blue cluster on the left, the Keurig

machine in turquoise on the top of the screen, the coffee cup in yellow in the top right, the plate in red in the middle, the K-cup as an orange dot in the center right, and our participants arm in pink on the bottom right.

Object Tracking

With object clusters defined in our video, we need a way to track when and where these objects move during the scenario. After all the objects and their locations are known in the initial video frame, object tracking can begin. As objects are moved about the table, we need to be able to follow where the objects are at all times. The events we are interested in depend on the continuous location of:

- the coffee cup
- the k-cup
- the Keurig machine's lid
- the toast
- the participant's hand

To aid in tracking, a number of preprocessing steps are performed on point cloud video

Pre-processing

The goal for pre-processing the collected video data is to filter out all static objects so that the moving objects that are of interest, the coffee cup, k-cup, etc. can more easily be tracked. In doing so, the processing time to track each object is improved as well. Since all of the objects of interest lie in a fixed region in front of the camera, a lot of depth points can be immediately thrown out. Points that are very close or very far from the camera such as the room's wall or other background imagery do not contribute any useful information and are filtered out in one pass through the input video bag file. Snapshots of the static objects can be found in the object clusters that make up the initial scene. The objects that should be removed are:

- the desk
- the toaster
- the plate

The desk can be removed by passing through every frame in video and applying the same rotation matrix that was used to level the desk and using the plane found previously to reject points. For the plate and toaster, points are rejected if they are sufficiently close to any of the points that make up their object clusters.

Tracking

After a video has been fully pre-processed, object tracking methods are applied to the objects that remain. The centroid of each object is computed from its initial cloud and is maintained through each frame as new object clouds are estimated. In each frame, each point is analyzed and a determination as to which object it belongs is made based on its location and hue. A quick check is made comparing the distance of the point to centroids of objects and for the objects that

this distance is reasonable, further checks are made. The closest point to each candidate object is found and a final determination is made based on the differences in hue. Specific parameters that describe the color spectrum for each object are also given to help make a distinction between ambiguous points. The parameters for each object will typically be the same but need to be tuned on a video by video basis. Once the objects are separated and tracked in the videos, we used their relative locations as inputs to our event understanding module.

NCP Context Triggers

As soon as objects are tracked, the state of the world that the NCP user was experiencing can be understood. Given the separate clouds that make up each object in every video frame, further analysis is done comparing the centroids of each cloud to determine when objects come near each other and an event trigger is fired when some state change has been detected. More specifically, the following event triggers occur in our scenario:

- coffee cup picked up
- coffee cup placed under Keurig machine
- K-cup picked up
- K-cup placed in Keurig machine
- Keurig machine lid `up'
- Keurig machine lid `down'
- Keurig machine button pressed
- bread picked up
- bread placed in toaster
- toaster lever down
- toast popped up
- toast placed on plate

The majority of triggers are identified by relative centroids of objects, a few cases however are found using specialized heuristics. For example, the toaster lever down trigger is fired when it is known that the bread is in the toaster and it can no longer be seen.

Event Understanding

Prof. Jeff Zacks, a consultant for the second year of NCP, has developed a model called Event Segmentation Theory (EST) that describes how humans segment ongoing continuous input from the environment into discrete events. In this task, we developed a simple computational version of EST and used it to provide understanding of the current state of events and the expectations for next stimuli that would occur. The advantage of doing this is that NCP always understands what is going on in the environment, and what normally should happen next. This has the effect of improving decisions made about user intent, since NCP knows to check for intent signals whenever a meaningful stimulus event occurs. In future versions, when the event stream includes

a variety of ongoing events, it will be possible to check for intent signals when event segmentation occurs, at the end of one event and at the beginning of another.

Events were described as hierarchical state machines, with external stimuli driving the transitions among states. There were two top-level events, Making Coffee and Making Toast. Embedded within each top-level event was two sub-events called Move Item to Device and Move Item to Table. In Figure 4 and Figure 5 below, states enclosed in boxes contain these sub-events.

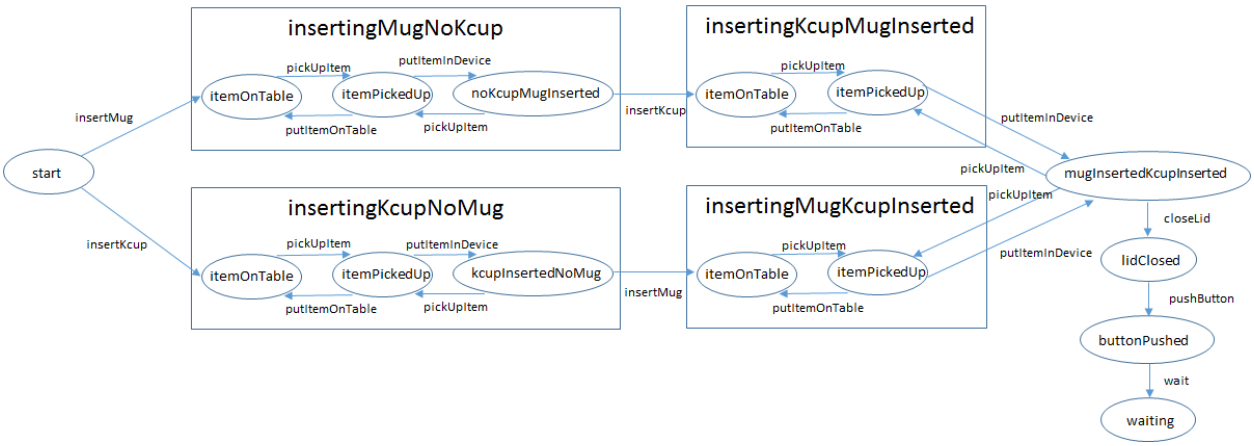

Figure 4: Hierarchical state machine events for making coffee.

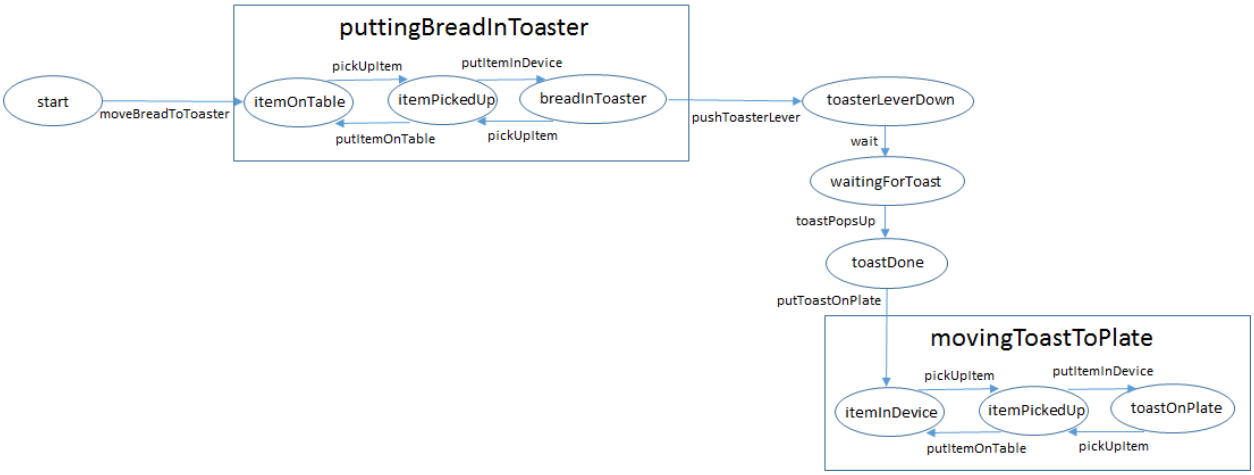

Figure 5: Hierarchical state machine events for making toast.

The NCP Event Understander is seeded with these events and sub-events. Presumably, in a larger system, such events and sub-events will reside in a library that can be automatically searched. Each event is identifiable by the first stimulus it is expecting. In NCP, these were unique; in a system with a larger event library, when events are no longer uniquely identifiable by their first expected stimulus, it will be possible to "run" all the matching events until the sequence of stimuli eventually results in a unique identification.

Once the event (or sub-event) is identified, it transitions to the next state whenever a legal next stimulus is observed. NCP handles the event or sub-event in a somewhat unusual way, one that corresponds fairly directly with Prof. Zack's empirical work on human event understanding called Event Segmentation Theory. [3,4] In particular, the mechanism for identifying an event boundary is that stimulus prediction begins to fail, signaling that "something new is going on." This is in contrast to a conventional hierarchical state machine where the machine simply reaches a terminating state. Finding event boundaries by prediction failure has a number of advantages: primarily that it allows one event to be interrupted by a new, "surprise" event, and, without any special changes to the event-handling mechanism, the new event can be processed the same way. In NCP, the prediction threshold was set at three consecutive stimulus prediction failures; when this happened, an event boundary was identified and the module looked for a new event based on the last three stimuli.

In NCP, while an event is running, a full description of the state of the top-level event and any sub-events is available, as well as a list of next "legal" stimulus values. This description is available any time. In future versions of NCP, it will be possible to query the Event Understanding module when a possible intent signal is identified in the EEG in order to help determine whether or not it was a genuine intent signal.

Figure 6 below gives a schematic overview of EST. (For more detail, see [5]) EST starts from the finding that sensory and perceptual processing is organized hierarchically, such that with each synapse from the sensory surface representations become spatiotemporally broader and more abstract. A crucial product of sensory and perceptual processing is a set of predictions about what will happen in the near future. Perceptual predictions are adaptive for the control of behavior, allowing one to perform anticipatory actions and thus to maximize outcomes. Perceptual predictions are guided by stable representations of "what is happening now," called *event models*. Event models are valuable because they allow the system to make predictions that are not disrupted by momentary occlusion or lapse of attention. Event models also can integrate information from long term memory through associative retrieval. In order for an event model to be useful, it needs to be stable throughout an event, but it also needs to be updated when the activity in the world transitions to a new event. How can people's cognitive system perform this updating, given that the world does not provide explicit event boundary cues? According to EST, event model updating is controlled by monitoring prediction error. When prediction error is low, one's current event model is likely appropriate to the circumstances and should be retained. However, when prediction error spikes the system should update the model. This updating is experienced as an event boundary.

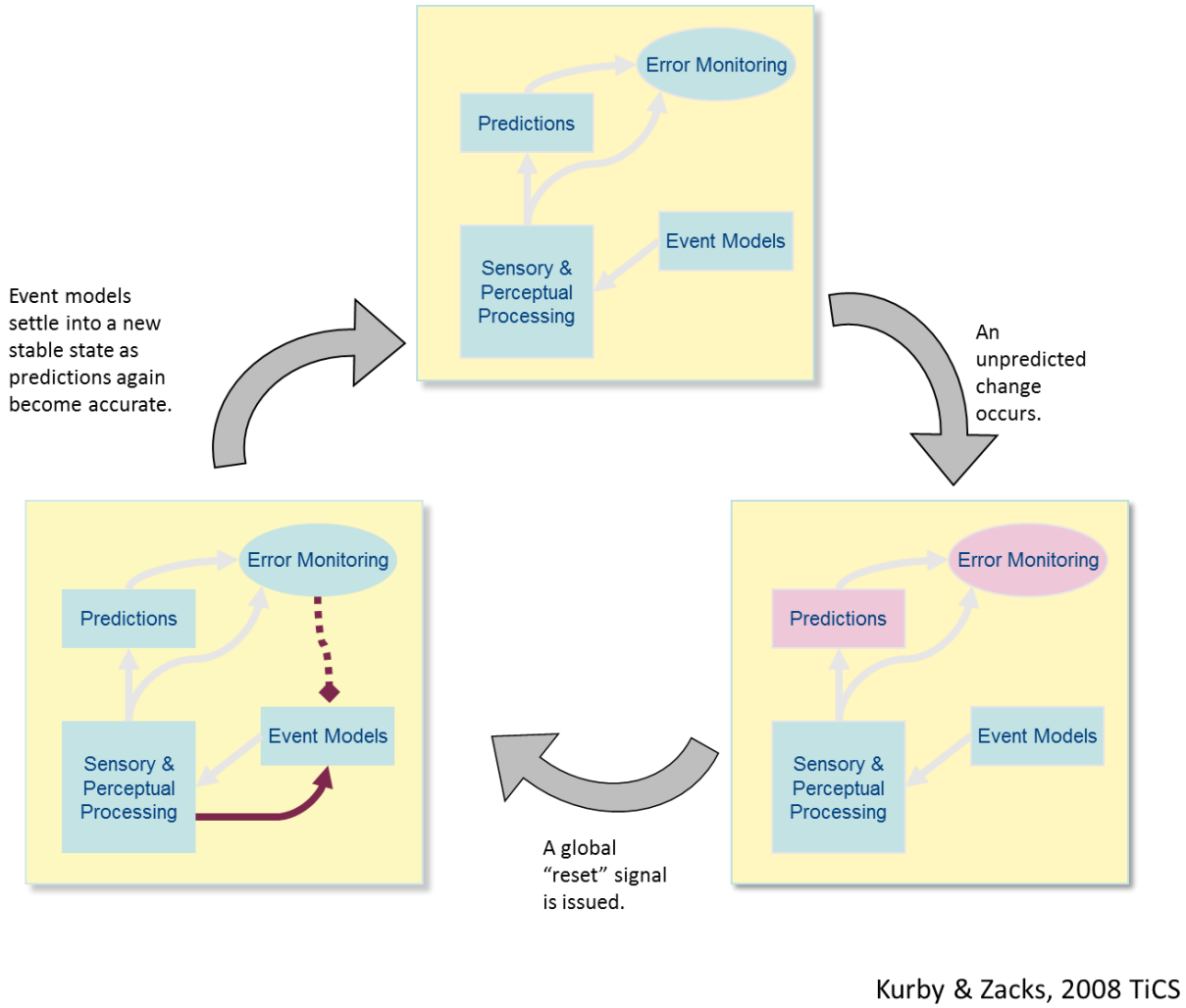

Figure 6: Zack's Event Segmentation Theory (EST).

One of the motivations for EST was the finding that observers tend to identify event boundaries when many features of the situation are changes. Feature changes associated with event boundaries include low-level features such as visual motion [6,7] and higher level features such as causes and goals [8]. Based on this observation, NCP will use feature changes in the task environment to identify event boundaries. These will be validated against observers' event boundary judgments.

In NCP, we made an effort to be faithful to EST insofar as possible, all the while understanding that eventually the NCP Event Understanding system can be extended to track multiple ongoing events, multitasking, and other kinds of event uncertainty that may be difficult for humans to manage. In future versions, several extension are possible:

- Event models may describe more general events using mechanisms like probabilistic context-free grammars or structured hidden Markov models.

- Stimuli might be based on more general action recognition routines, or even on primitive event detections.
- Event models could be adapted with experience, or even learned from scratch.
- Model identification could be based on probabilistic matches, and feasible model candidates could be "run" until the correct model is identified.
- Probabilistic prediction failures could be used.
- Better recovery from random prediction errors could be implemented.

Analysis

EEG Analysis

Using the EEGLAB toolbox from the Schwartz Center for Computational Neuroscience out of the University of California San Diego [9], EEG data were further filtered with a low pass filter set at 10 Hz. According to prior similar work [10, 11, 12], we chose to focus our search for an intent signal on the following channels, located primarily toward the top (or vertex) of the head: C1, Cz,, C2, CP1, CPz, and CP2.

The data were segmented into 200 ms windows (including 100 data points in each window). For each window, we calculated the minimum amplitude, the maximum amplitude, and the maximum slope (second derivative of the signal). These features were then used to train a linear discriminant analysis (LDA) machine learning method. We trained the LDA using labeled data that either belonged to an 'intent' class or a 'no-intent' class (based on EEG markers indicating actions vs. non-action times). The data that belonged to the intent classes came from data windows that occurred 1.5 seconds before to 1 second after an observed movement in the corresponding video (green segments in Figure 7 below). The training data for the no-intent classes came from data that had no movement in the video (red segments in figure below). The red vertical lines in the figure below indicate the observed movements of the participant in the video.

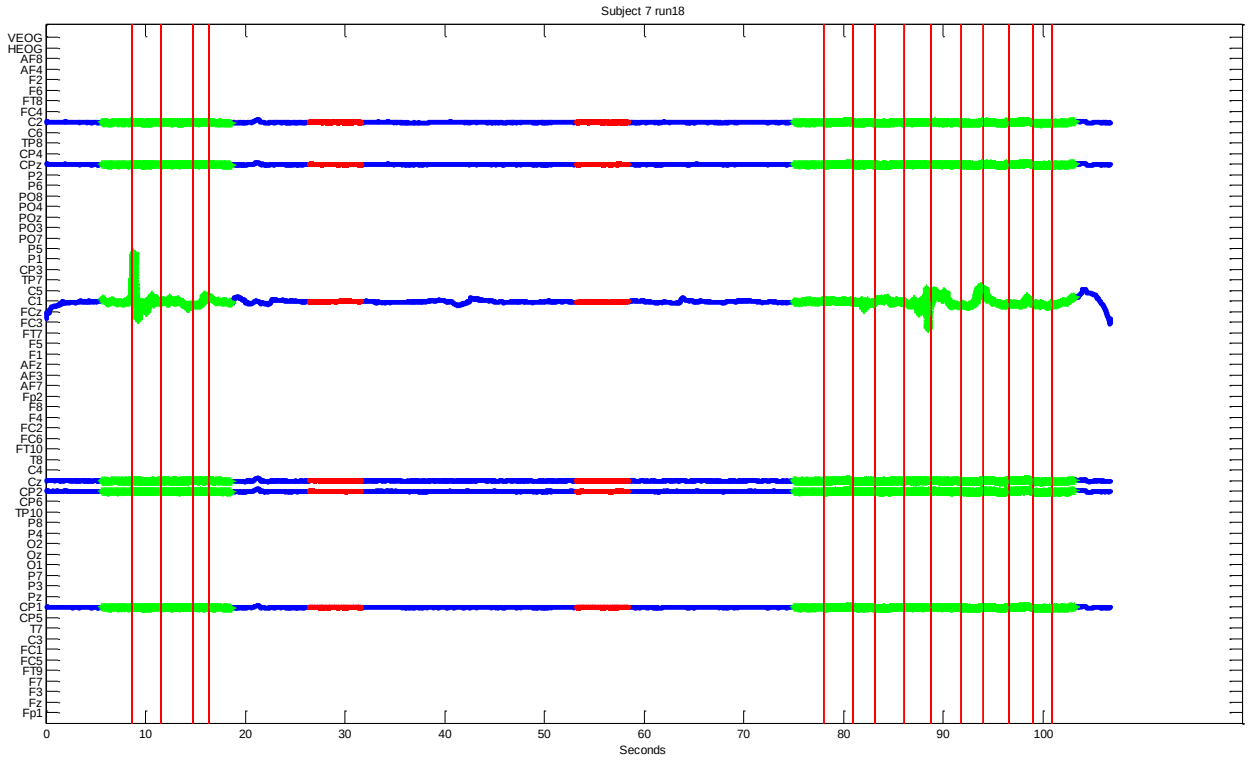

Figure 7: Training data for one run of data collection. The red vertical lines are the places where movement was started in the video. The green segments indicate the training data for the intent signal, while the red segments are the training data for the no-intent signal.

The output of the LDA was a probability value for each data point belonging to each class over time. Figure 8 below illustrates the probability of that data point belonging to the 'intent' class in blue and the probability of belonging to the no-intent class in red. The first and third segments (starting from the far left) are no-intent segments, while the second and fourth segments are intent segments. While our classification is not perfect, you can see that on average, there are more red points towards the top of the figure (higher probability) in the first and third segments, while there are more blue points towards the top in the second and the fourth segments.

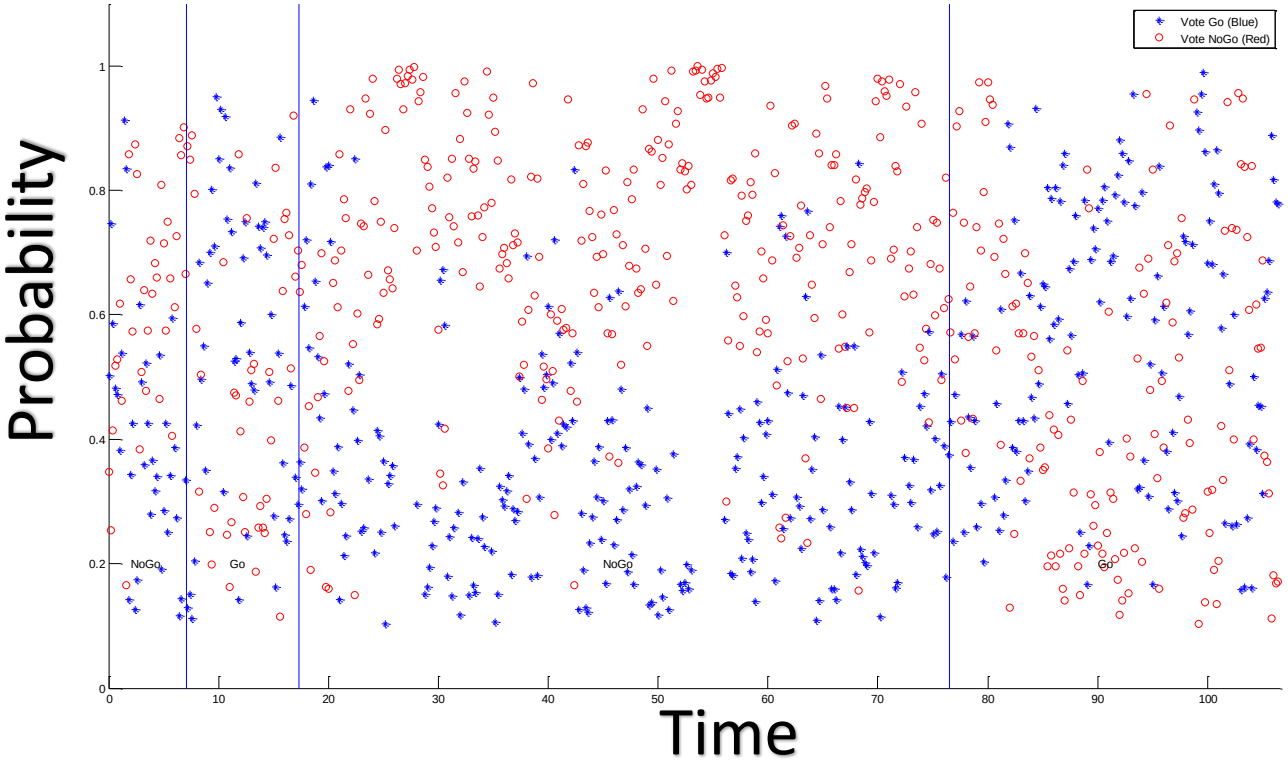

Figure 8: The output of our Linear Discriminant Analysis (LDA) classifier where red dots indicate the probability of no-intent, and blue crosses are the probability of intent.

The classification rate can be calculated using a ROC curve. In statistics, a receiver operating characteristic (ROC), or ROC curve, is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. The curve is created by plotting the true positive rate against the false positive rate at various threshold settings. The area under the ROC curve is a measure of how accurate you can expect your classifier to be. The ROC curve we calculated is shown in Figure 9 below and the area under the curve (AUC) is 0.73991.

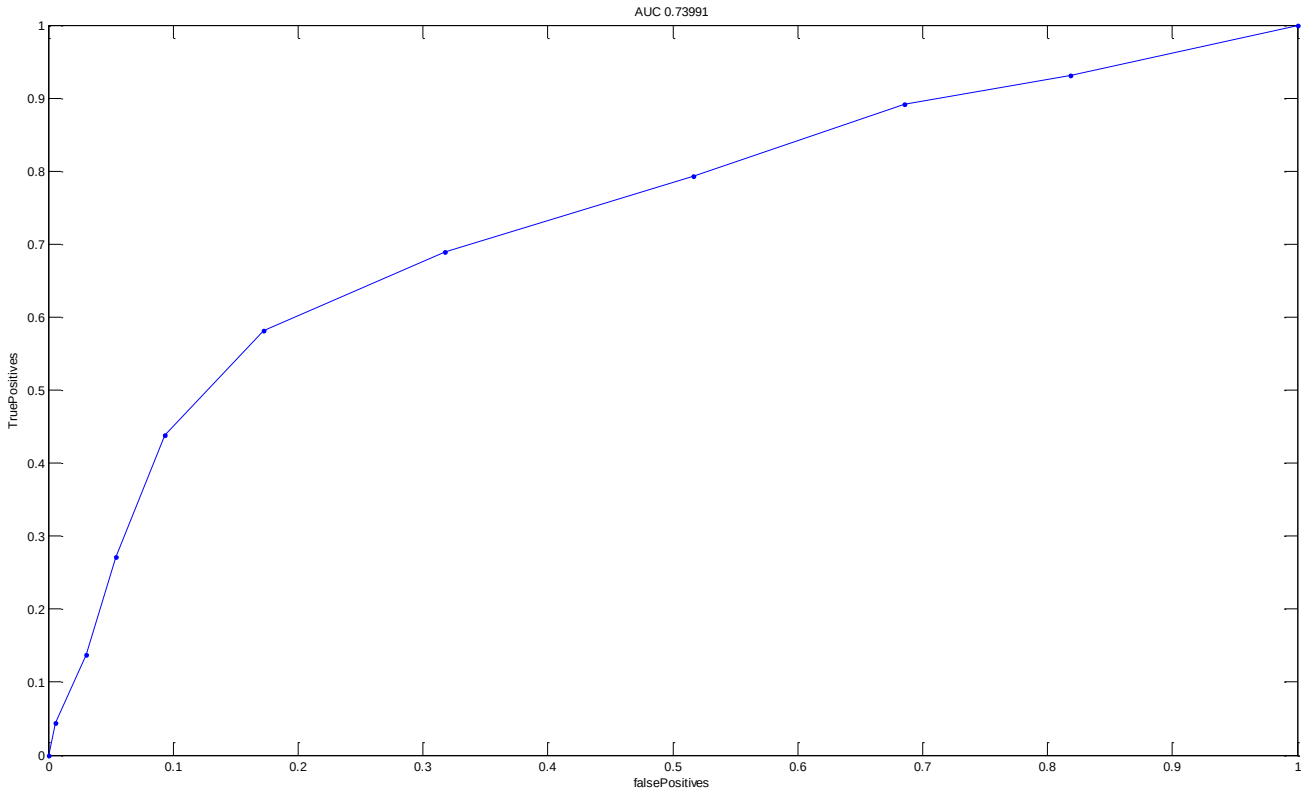

Figure 9: The ROC curve for our LDA classifier with an AUC of 0.73991.

The output of the EEG Analysis is used along with the Event Understanding module to prompt the action of our robotic limb. When a trigger is detected in the object recognition and tracking module, the EEG analysis is queried to see if there was a probable intent signal. If so, an action is selected from the Event Understanding module and sent along to our action NCP action module.

NCP Action Module

The NCP Action Module receives the most probable action from the Event Understanding Module and translates it as an enumerated code that is mapped to a file that is understandable by the JACO robotic arm. A set of waypoints are stored in this file in which the arm can traverse to carry out the intended action. Upon receiving the signal, the arm will follow the designated trajectory and notify the system when the last waypoint is reached.

JHU/APL was responsible for developing a programmatic interface to a robotic manipulator that would allow the system to perform autonomous manipulation tasks such as making a cup of coffee or toast, simulating an activity of daily living that could be performed by the robot. As part of this effort, JHU/APL developed a control interface to the Kinova JACO arm, a wheelchair mountable, 6-DOF upper arm and 3 finger gripper (Figure 10) that has the precision necessary to

complete the tasks outlined in this project. We also developed a waypoint interface that provides an intuitive, high-level interface for controlling a Kinova JACO manipulator arm. It allows users to create spatial waypoints for the JACO end effector and generate trajectories between the waypoints. The waypoints can also be grouped together to enable more complex motions. In addition, waypoints can be saved to a JSON file to be loaded in the future, and each waypoint can be transformed into an arbitrary frame of reference allowing for more flexibility in the reuse of saved trajectories. After developing this software, we used it to perform two tasks with the JACO arm: making toast and making coffee.

Figure 10: Kinova JACO arm

The typical workflow while using the interface consists of moving the JACO arm to an initial position, selecting a reference frame, and creating a waypoint at that location. The user then steps the arm through a series of points along its desired trajectory and creates a waypoint at each point. After a set of waypoints has been built, the user creates one or more groups and adds each waypoint to one of the groups. Once set up, the user can then use the interface to generate a trajectory for an entire group of waypoints and have the arm execute the trajectory. After testing

the groups, the user will finally use the interface to save the groups of waypoints to a JSON file so that they can be reloaded later.

The architecture of the Waypoint Interface is designed to be modular and allow for new functionality to be added easily (see Figure 11). On the user side, the interface can be operated through one of several modules. The most user-friendly is the front-end GUI, which also provides the most functionality to the user. In addition to the GUI, a ROS Action Server API and a TCP Server API are also provided. Both servers can load waypoint files, generate waypoint trajectories, and move the gripper, but do not provide the more advanced functionality offered by the GUI, such as saving files or adding new waypoints.

In addition to the modules exposed to the user, the Waypoint Interface contains several internal modules, each of which manages a subset of its functionality. The Controller Manager module interfaces with the external motion planning node to generate and execute trajectories, while the Gripper Manager module handles the opening and closing of the JACO arm's fingers. Finally, the Marker Manager module sends visualization data to RVIZ so that the user can view waypoint locations in real time.

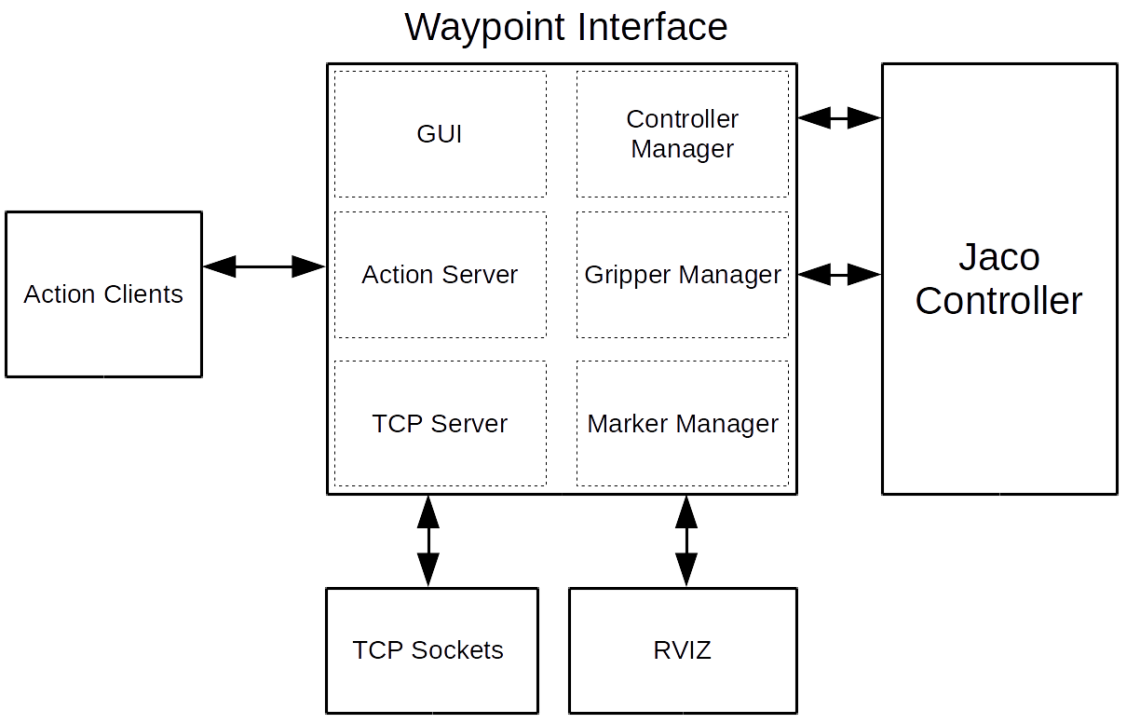

Figure 11: Software Architecture

Using the Waypoint Interface, we developed two action sets: one for making toast, and another for making coffee. For the toast making scenario, we assumed a fixed transform between the JACO arm and the toaster, and that the untoasted bread was already in the toaster. Therefore, the arm only needed to press the lever to begin toasting and then pick up the bread from the toaster. We divided the toast-making task into the following subtasks: "push_toaster" (pressing the lever on the toaster), "toast_ready" (picking up the toast from the toaster), and "grab_toast" (placing the toast on the table).

The second task, making coffee, was more challenging. The design of the coffee maker was such that the positioning of the K-cup required millimeter accuracy to fit properly (Figure 12). To solve this problem, we developed careful calibration procedures with visual servoing to ensure that we could reproducibly pick up the K-cup and place it in the correct location. We divided the coffee-making task into the following subtasks: "pickup_kcup" (picking up the K-cup from the table), "place_kcup" (placing the K-cup into the Keurig coffee maker), "close_lid" (closing the lid of the coffee maker), and "push_button" (pressing the Start button on the coffee maker). These string based commands provide a high enough level of abstraction to allow Aptima to integrate with the manipulator without requiring low level knowledge about the robotic manipulator and its kinematic and dynamic properties.

Figure 12: Insert K-Cup into coffee machine

Results

The result of this project was that we were able to create a system that used object recognition to feed a hierarchical state machine which segmented the world into events which helped to trigger the timing of EEG analysis. The output of this analysis is a probability of motor intent, which if high enough, triggers the start of a context-appropriate action on the robotic arm that was selected as the most likely action from the event understanding module.

Conclusions

NeuroCognitive Patterns allows a user to control a robotic arm through a brain computer interface. NCP uses EEG from the user to detect probable motor intent signals with no special user effort required. NCP also uses contextual knowledge about the environment from video and depth information as well as an event library to identify the most likely times for a user action and the most likely action at those times.

This result, while still preliminary, does show that there is promise for context driven neural control of automation. This will result in a new generation of automation that will require dramatically less attention from the operator, freeing the operator to attend to other mission-important tasks.

Recommendations

The NCP prototype developed on this project can be enhanced in several meaningful ways, and we recommend that they be explored in future efforts:

- The event library for the event understanding system should be dramatically expanded, preferably by automatically learning event schemata.
- By expanding the capabilities of the object recognition and tracking, event understanding, and action capabilities, an NCP system capable of operating in more contexts will be possible. These capabilities current depend on the prototype context.
- Integrating sensors on the prosthetic arm itself will allow for mobile use of NCP. It may be necessary to use sensors other than Kinect, such as Intel's RealSense.
- To be usable by real patients, NCP will need to infer intent from EEG in real time.

Proposed Additional SBIR/STTR Funded Research

Background

Neurocognitive Patterns (NCP) is an unobtrusive neural prosthetic interface that uses shared context to improve the accuracy and specificity of predictions of user intent. A key part of the

architecture that enables NCP to do this is the Event Understanding Module (EUM). The EUM is inspired by Zacks' Event Segmentation Theory (EST; Kurby & Zacks, 2008; Zacks, Speer, Swallow, & Maley, 2010) that describes how humans segment ongoing continuous input from the environment into discrete events. EST holds that for each perceived event, humans identify a predictive event model and use it to predict the stimuli it should encounter next. When those predictions begin to fail, humans draw an event boundary, identify the next event, and begin making new predictions.

The EUM works this way as well: incoming stimuli are used to identify a model for the current event, and that model stays current until its predictions begin to fail. In the EUM, events are described as hierarchical state machines (HSMs), with external stimuli driving the transitions among states. The EUM is used to provide the system with understanding of the state of the current events and expectations for next stimuli that will occur.

Since the prototype of NCP was limited in the range of events it could understand, the HSMs in the EUM were hand-crafted. In order to develop a more general event-understanding capability, though, it will be necessary to use a more general event model structure, and to learn a large number of event models using that structure.

Learning both the structure and the models themselves is feasible with modern machine learning techniques, most notably probabilistic models of inductive learning. [13, 14, 15] These techniques focus on learning knowledge structures as well as the knowledge itself. Notably, such approaches can learn knowledge from a relatively small amount of data (much as humans can), because the learned structure of the knowledge serves as a strong prior basis for the form the knowledge can take. This is in contrast to most deep learning approaches that require massive amounts of data.

Approach

In the NCP follow-on effort, the team will develop techniques for learning the structure and content of many event models, with the goal being to use them to drive action and understanding in training scenarios in simulated environments. To accomplish this, the team will

- Develop use case(s)
  - Select one or more domains of interest
  - Identify corpora and primitive analysis software (such as computer vision routines)
- Perform research to understand how event models need to change when the event perceiver is also a participant in the event, that is, for models of active events. Current event models, including EST, only account for event understanding when the event perceiver's actions are not part of the event, that is, for models of passive events

- Develop and test automated structure-and-knowledge acquisition routines for active and passive event models
- Automatically learn active and passive event model structure and event models for use case domains
- Develop a prototype to learn and use the event models
- Coordinate with relevant ONR MURIs, including those concerned with Episodic Memory and Event Representation and with Interactive Task Learning

Transition and Acquisition Planning

The capabilities developed under NCP can be used in a variety of contexts. One such context that will have likely transition paths is to expand the library of event modules dramatically by automatically learning and using them. Such event models and event understanding and execution capabilities can be used in scenario-based training to provide students with intelligent, event-aware computer generated forces. This will reduce the current training support burden for human training professionals.

References

1. M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *Proc. Open-Source Software workshop of the International Conference on Robotics and Automation (ICRA)*, 2009.

2. T. Cover and P. Hart. "Nearest neighbor pattern classification*." IEEE Transactions on Information Theory* **13** (1): 21–27, 1967.

3. Kurby & Zacks, "Segmentation in the perception and memory of events." *Trends in cognitive sciences* 12:72-79, 2008.

4. Zacks, J. M., Speer, N. K., Swallow, K. M., & Maley, C. J. *"*The brain's cutting-room floor: Segmentation of narrative cinema*." Frontiers in Human Neuroscience, 4:*1–15, 2010.

5. Zacks, J.M., Speer, N.K., Swallow, K.M., Braver, T.S., and Reynolds, J.R. "Event perception: a mind-brain perspective." *Psychological Bulletin.* 133:273-293, 2007.

6. Zacks JM. "Using movement and intentions to understand simple events." *Cognitive Science.* 28(6):979–1008, 2004.

7. Zacks, J. M., Kumar, S., Abrams, R. A., and Mehta, R. "Using movement and intentions to understand human activity." *Cognition* 112:201–216, 2009.

8. Swallow KM, Zacks JM, Abrams RA. "Event boundaries in perception affect memory encoding and updating." *Journal of Experimental Psychology: General*. 138:236–257, 2009.

9. A Delorme & S Makeig "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics." *Journal of Neuroscience Methods* 134:9-21, 2004.

10. Jankelowitz SK, Colebatch JG. "Movement-related potentials associated with self-paced, cued and imagined arm movements." *Exp Brain Res.* 147: 98–107, 2002.

11. Lew E., Chavarriaga R., Silvoni S., Millán J. D. R. "Detection of self-paced reaching movement intention from EEG signals." *Front. Neuroeng.* 5:13, 2012.

12. Planelles D., Hortal E., Costa A., Ubeda A., Iáez E., Azorín J. M. "Evaluating classifiers to detect arm movement intention from EEG signals." *Sensors (Basel)* 14:18172–18186, 2014.

13. Tenenbaum, J.B., Kemp, C., Griffiths, T.L. & Goodman, N. "How to grow a mind: Statistics, structure, and Abstraction." *Science*, **331**, 1279-1285, 2011.

14. Griffiths, T.L., Yul, E., & Sanborn, A.L. "Bridging levels of analysis for probabilistic models of cognition." *Current Directions in Psychological Science*, **21**(4), 263-268, 2012.

15. Lake, B.M., Ullman, T.D., Tenenbaum, J.B., & Gershman, S.J. "Building machines that learn and think like people." *CBMM Memo No. 046.* Retrieved from https://cbmm.mit.edu/sites/default/files/publications/machines_that_think.pdf, 2016.